Hindawi Publishing Corporation
Journal of Sensors
Volume 2016, Article ID 1547963, 9 pages
http://dx.doi.org/10.1155/2016/1547963

Research Article

A Lightweight Authentication and Key Management Scheme for Wireless Sensor Networks

Danyang Qin, Shuang Jia, Songxiang Yang, Erfu Wang, and Qun Ding

Key Lab of Electronic and Communication Engineering, Heilongjiang University, Harbin, China

Correspondence should be addressed to Danyang Qin; qindanyang@hlju.edu.cn

Received 3 June 2016; Revised 6 September 2016; Accepted 21 September 2016

Academic Editor: Jose Maria de Fuentes

Security problem is one of the most popular research fields in wireless sensor networks for both the application requirement and the resource-constrained essence. An effective and lightweight Authentication and Key Management Scheme (AKMS) is proposed in this paper to solve the problem of malicious nodes occurring in the process of networking and to offer a high level of security with low cost. For the condition that the mobile sensor nodes need to be authenticated, the keys in AKMS will be dynamically generated and adopted for security protection. Even when the keys are being compromised or captured, the attackers can neither use the previous keys nor misuse the authenticated nodes to cheat. Simulation results show that the proposed scheme provides more efficient security with less energy consumption for wireless sensor networks especially with mobile sensors.

1. Introduction

Wireless sensor networks (WSNs) [1] consist of a large number of nodes in a self-organized manner, where there are no central control nodes, and the nodes lying out of the transmitting range can communicate in a multihop way. As the wireless sensor network is independent of the predeploy infrastructure, it has broad application prospects in the battlefield environment, disaster relief, and environmental threats exploration, which make the security and efficiency the most basic requirements and the most popular research areas [2].

The characteristics of wireless sensor networks determine the network security threats, the security systems, and security algorithms that are quite different from those in traditional networks [3], and the traditional network security systems and security algorithms cannot be introduced directly. Meanwhile, the inherent essence of limited storage space, computational capabilities [4], bandwidth, and communication energy does not make the computational data encryption and public key cryptography based on the traditional cryptographic techniques adapt to wireless sensor networks. The security system and algorithm for WSN are mainly focused on in this paper to design an effective Authentication and Key Management Scheme with low computing and energy cost.

2. Related Work

With the development of security technology in wireless sensor networks, the research on routing protocols has been increasing in recent years. This section describes the three existing master key-based key management protocols: LOCK [5], SPINS [6], and BROSK [7]. These protocols have been widely discussed in this area.

The Localized Combinatorial Keying (LOCK) proposed by Eltoweissy is an Exclusion-Based Systems (EBS) dynamic key management approach for cluster-based sensor networks. LOCK takes use of three keys, including the administrative key, the group session key, and the cluster session key. A special node selected by the cluster head is called a key generation node and will perform a key generation process. LOCK is for static networks. But the proposed scheme in this paper will be suitable for dynamic networks.

SPINS is a famous security framework for wireless sensor networks. Although it contains two protocols, SNEP and TESLA, which are used to achieve the confidentiality and

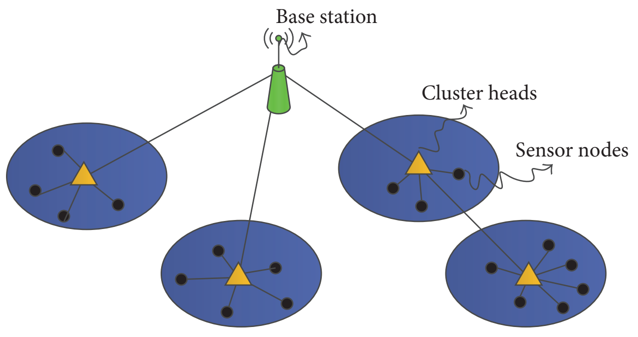


Figure 1: Wireless sensor network model.

authentication of data broadcasting, respectively, we will concentrate on the key agreement protocols [8].

BROSK can be considered as a more recent ad hoc key agreement protocol compared to SPINS. There are no trusted parties or servers in this scheme, in which each node negotiates the session key directly with its neighbor node by broadcasting key agreement message.

3. Network Model and Key Generation

This part briefly introduces the network model, the importance of authentication, and the idea of CPK system based on ECC.

3.1. Network Model. The members of wireless sensor network are BS (base station), CH (cluster head), and resource-constrained nodes, which are deployed in a geographical area to perform some special monitoring functions. In most applications, especially for large scale deployment, the sensors are arranged in multiple static clusters, as shown in Figure 1. The members' changing makes the authentication and key management always a key research point in wireless network. Considering the resource-constrained essence of WSN, a lightweight scheme is badly needed, which keeps the key changeless to save the limited energy. Many applications, however, require the mobility of network nodes to support. In such a mobile sensor network, there will be always the condition that a node from an existing cluster moves into another cluster. The separated nodes may be the cluster heads or cluster members. The main reason causing the changes in cluster heads and cluster members is the mobility. The mobility of nodes together with the transient nature of the wireless media often leads to a highly dynamic network topology. In this case, security protection with moving sensors must be incorporated into wireless sensor network.

Authentication is one of the security practices to verify the identity of the sensor nodes. Public key cryptography is a popular way to provide authentication for WSN. Though the easy design and effective operation make it attractive, the disadvantage of more energy requirement greatly restricts the network performance. Thus, the sensor node has to use elliptic curve digital signature algorithm to generate a digital signature authentication. The combination of pairwise, global key, cluster key, and preloaded secret information is also used to verify the sensor nodes in the network. The node mobility will lead to random topology changes, which affects the security of mobile sensor networks. The lightweight Authentication and Key Management Scheme (AKMS) proposed in this paper will adopt a Hash Message Authentication Code (HMAC) algorithm [9] and a Combined Public Key (CPK) password system based on ECC to authenticate the moving nodes within the network effectively.

Sensor network has many features that make them more vulnerable to attack than the traditional computing devices. For example, the nature of the broadcasting allows the information to be intercepted, eavesdropped on, tampered with, or exchanged easily. Besides suffering the same threats with the conventional wireless networks, WSN is vulnerable to resource depletion attacks, which attempt to run out of resources, such as node battery and network bandwidth, and causes more damage. Finally, most devices in WSN cannot be tamper-resistant typically, which facilitates the physical manipulation and keys being stolen. To approach the real condition, the attack model is assumed as follows:

(1) The sensors are not tamper-resistant, so an attacker is able to access the stored information and the keys in the node storage directly.

(2) The attackers may appear not only before the network deployment but also during all the network life cycle without any assumptions about the quantity or the physical location of the attackers.

(3) The attackers may easily intercept and modify the exchanged information among the network nodes.

3.2. The Idea of CPK System Based on ECC. Combined Public Key (CPK) password system based on ECC is a way of authentication based on the identity. According to the mathematical principle of elliptic curve discrete logarithm, we build public key matrix and private key matrix and use the hash function to map the entity's identity for the row and column coordinates sequence of the matrix; it is used for the selection and combination of matrix element, and it can generate a large number of public and private key pairs, so as to realize the large scale of identity-based key generation and distribution. Entity nodes need to know each other's identity to calculate its public key, which can easily achieve authentication and security features. Among them, identify key is generated by the entity's identity through combination matrix. The CPK system based on ECC has the following advantages.

(1) In wireless sensor network (WSN), the only legitimate nodes have the private key, and, according to the other identity ID and segmentation key, we can calculate the other Combined Public Keys (CPK), so the simple and efficient authentication process can be realized without the participation of the third party.

(2) The CPK system based on ECC can combine large public/private key pair through a small amount of public/private key matrix; nodes only need to store a small matrix to achieve a large number of nodes' security authentication in the network.

Point multiplication operation is the foundation of CPK algorithm. ECC Signature Algorithm (ECDSA) is the elliptic curve version of digital signature algorithm (DSA); it is the basis for the CPK digital signature algorithm. This paper adopts ECC algorithms based on Montgomery type curve [10]. We use point multiplication operation of Montgomery type elliptic curve and the binary shift NAF coding algorithm to solve the large amount of calculation generated by ECC point multiplication. We use the point addition and times point fast operation where the value of y is not calculated to avoid modular inversion algorithm under the projective coordinates.

Point addition formula is as follows:

$$\begin{aligned} X_{m+n} &= Z_{m-n}\left[(X_m - Z_m)(X_n + Z_n) + (X_n - Z_n) \cdot (X_m + Z_m)\right]^2 \\ Z_{m+n} &= X_{m-n}\left[(X_m - Z_m)(X_n + Z_n) - (X_n - Z_n) \cdot (X_m + Z_m)\right]^2 . \end{aligned} \quad (1)$$

Times point formula is as follows:

$$\begin{aligned} 4X_nZ_n &= (X_n + Z_n)^2 - (X_n - Z_n)^2 \\ X_{2n} &= (X_n + Z_n)^2 (X_n - Z_n)^2 \\ Z_{2n} &= (4X_nZ_n)\left[(X_n - Z_n)^2 + \left(\frac{(A+2)}{4}\right) 4X_nZ_n\right]. \end{aligned} \quad (2)$$

Calculating coordinates (X, Z) of the times point dP of point $P = (x, y)$ in projective coordinates, where "$\leftarrow$" denotes mapping and (S1) means step 1, the specific algorithm is as follows:

(S1) $i \leftarrow |d| - 1$;

(S2) Calculate the integer:

$$\begin{aligned} X_1 &\longleftarrow x, \\ Z_1 &\longleftarrow 1; \\ T_1 &\longleftarrow (X_1 + Z_1)^2 - (X_1 - Z_1)^2 \\ X_2 &\longleftarrow (X_1 + Z_1)^2 (X_1 - Z_1)^2 ; \\ Z_2 &\longleftarrow T_1\left((X_1 - Z_1)^2 + \left(\frac{(A+2)}{4}\right) T_1\right); \end{aligned} \quad (3)$$

(S3) If $i = 0$, then jump to (S12), else go to (S4);

(S4) $i \leftarrow i - 1$;

(S5) If $d_i = 0$, then go to (S6), else jump to (S9).

(S6) Calculate the integer:

$$\begin{aligned} T_1 &\longleftarrow X_2; \\ X_2 &\longleftarrow \left[(T_1 - Z_2)(X_1 + Z_1) + (T_1 + Z_2)(X_1 - Z_1)\right]^2 \\ Z_2 &\longleftarrow x\left[(T_1 - Z_2)(X_1 + Z_1) - (T_1 + Z_2)(X_1 - Z_1)\right]^2 ; \end{aligned} \quad (4)$$

(S7) Calculate the integer:

$$\begin{aligned} T_1 &\longleftarrow X_2; \\ T_2 &\longleftarrow (T_1 + Z_1)^2 - (T_1 - Z_1)^2 \\ X_1 &\longleftarrow (T_1 + Z_1)^2 (T_1 - Z_1)^2 ; \\ Z_1 &\longleftarrow T_2\left((T_1 - Z_1)^2 + \left(\frac{(A+2)}{4}\right) T_2\right); \end{aligned} \quad (5)$$

(S8) Jump to (S3);

(S9) Calculate the integer;

$$\begin{aligned} T_1 &\longleftarrow X_1; \\ X_1 &\longleftarrow \left[(X_2 - Z_2)(T_1 + Z_1) + (X_2 + Z_2)(T_1 - Z_1)\right]^2 \\ Z_1 &\longleftarrow x\left[(X_2 - Z_2)(T_1 + Z_1) - (X_2 + Z_2)(T_1 - Z_1)\right]^2 ; \end{aligned} \quad (6)$$

(S10) Calculate the integer:

$$\begin{aligned} T_1 &\longleftarrow X_2; \\ T_2 &\longleftarrow (T_1 + Z_2)^2 - (T_1 - Z_2)^2 \\ X_2 &\longleftarrow (T_1 + Z_2)^2 (T_1 - Z_2)^2 ; \\ Z_2 &\longleftarrow T_2\left((T_1 - Z_2)^2 + \left(\frac{(A+2)}{4}\right) T_2\right); \end{aligned} \quad (7)$$

(S11) Jump to (S3):

(S12) Output integer X_1, Z_1, as dP corresponding X, Z.

4. Proposed Authentication and Key Management Scheme

The lightweight AKMS proposed in this paper consists of three main phases: key predistribution phase, network initialization phase, and authentication protocol. The first phase is enabled before the nodes are being deployed. The second phase sets the security of network, and it is enabled during the network deployment. The last phase is enabled when a new node joins the network with the previous stage being over.

4.1. Key Predistribution Phase.

Key predistribution phase is a key step for dynamic key management with moving nodes in WSN. For reasons of clarity, the symbols used in this paper are listed in the Notations.

In this phase, a network-wide symmetric master key will be generated and stored securely. This key should be long enough to destroy the common attack, namely, a minimum of 128 bits. During the networking stage, each node is preinstalled with an initial authenticator. The ith cycle authenticator ∇^i can be used by a node to identify another node, the superscript symbol of which indicates the cycle where the authenticator takes. It consists of the random number of n tuples and the results of using a keyed-hash function with the current authentication key over them.

During the first authentication cycle, the authentication key is equal to the master key $k^0_{\text{auth}} = k_M$ before the deployment; therefore

$$\nabla^0 = \left\{\left(r_i, [r_i]_{k_M}\right)\right\}, \quad i = 0, \dots, n-1. \tag{8}$$

In general, the authentication key of the first cycle is $k^j_{\text{auth}} = [k_M]^j$; then the authenticator set is

$$\nabla^j = \left\{\left(r_i, [r_i]_{k^j_{\text{auth}}}\right)\right\}, \quad i = 0, \dots, n-1; \tag{9}$$

when the tuples are exhausted at this time, the authenticator will transmit to the next cycle.

4.2. Network Initialization Phase. This phase is enabled during the network deployment. In such operating environment, each node can find its neighbors within the communication range. Specific steps are as follows:

(1) Each node i generates its unique symmetric key by the CPK system based on ECC, k^i_{enc}, called the node encryption key, which is obtained by generating a random number and performing $k^i_{\text{enc}} = [k_M, r_i]$. For example, the encryption key of some node A can be calculated as $k^A_{\text{enc}} = [k_M, r_A]$.

(2) For a very short time, each node broadcasts its random value r_i with the unit as seconds [11]. In this way, the attackers listening to the broadcast communication will get the random values.

(3) Each node receives a random value from its neighbor node and uses common master key to calculate their encryption key. In this case, each node will store a list of paired keys of its neighbor nodes.

(4) Each node hashes the common master key and keeps it with the first forms of authentication key as $k^1_{\text{auth}} = [k_M]$, for the easy reason that storing master key in node's storage space has great potential danger if a node is captured. This is mainly because of the existence of the authenticator, which will help to authenticate other nodes and to verify the information of common master key without storing master key.

(5) In this stage, each node stores its encryption key k^i_{enc}, the set of encryption keys of its neighbor nodes, and the keys of the next authentication cycle k^1_{auth}, which is hash function of the master key and the current authenticator, ∇^0 consisting of the set of n tuples.

(6) Now, the node begins to communicate with other nodes using the encryption key in pair.

4.3. Authentication Operator. Authentication operation is used for network nodes to authenticate each other. The operator's goal is to provide the ability to verify the new node in the network once the deployment phase of nodes is over. Retaining the master key in the node internal memory could cause the whole network security to be damaged. To avoid this situation, AKMS precalculates the necessary authentication material (excitation/response tuple) and the authentication key which will be deleted later. Therefore, the new node will be verified by the knowledge of the authentication key without being stored in the memory. As mentioned above, the authentication operation will adopt two encryption primitives, as the excitation/response scheme [12] and the key chain [13].

4.3.1. Authenticator Generation. The authentication of any cycle j is constructed from the keys of the previous cycle $j-1$. In this way, the node can verify the master key, because the authentication key of the cycle $j-1$ can only be derived rather than storing the master key itself. If a node is destroyed, the attacker will only obtain the identity of the current cycle but not compromise the authentication and the exchange of keys performed using the previous cycles of the authenticator. As a node runs out of authenticators' instances, it will simply generate a new set; that is, it will start a new cycle of the authenticators set. The cycle should be composed of the following steps:

(1) A new authenticators set with n tuples of random numbers is calculated and the current authentication key k^j_{auth} is applied to each of them to obtain

$$\nabla^{j+1} = \left\{\left(r_i, [r_i]_{k^j_{\text{auth}}}\right)\right\}, \quad i = 0, \dots, n-1. \tag{10}$$

(2) The current authentication key is updated and hashed to obtain

$$k^{j+1}_{\text{auth}} = \left[k^j_{\text{auth}}\right]. \tag{11}$$

(3) The new key is generated by

$$\left[k^{j+1}_{\text{auth}}, \nabla^{j+1} = \left\{\left(r_i, [r_i]_{k^j_{\text{auth}}}\right)\right\}\right], \quad i = 0, \dots, n-1. \tag{12}$$

4.3.2. Implementation Issues. Each of these tuples has a status label to describe its current status in the authentication process. The possible values of status are as follows:

UNUSED: the tuple is unused

ASSIGNED: the tuple is temporarily assigned to a node in an ongoing process of authentication; if the process fails, the label will change to UNUSED, and the tuple can be used again

USED: the tuple is used in a successful authentication process, which cannot be adopted by any other processes; in this way, the replay attack will be avoided efficiently

In addition to these labels, there is another domain in authenticator structure, called the current tuple index denoted as δ to store the first UNUSED tuple, which will

Table 1: Example of the authenticator with $\delta = 3$.

Authenticator ∇^2	Status label of each tuple
$\left(r_0, [r_0]_{k^2_{\text{auth}}}\right)$	USED
$\left(r_1, [r_1]_{k^2_{\text{auth}}}\right)$	USED
$\left(r_2, [r_2]_{k^2_{\text{auth}}}\right)$	ASSIGNED
$\left(r_3, [r_3]_{k^2_{\text{auth}}}\right)$	UNUSED
$\cdots$	$\cdots$

increase when each tuple changes its status from UNUSED to ASSIGNED.

Finally, another important issue to be analyzed is the size of the authenticator n. As a system parameter, the value of n should be carefully set according to the number of nodes and the moving rate. Of course, if the network status is changing dramatically, the value of n will be adjusted dynamically. A typical value of n is about 10, which is relatively reasonable taking into account the expected authentication rate and the number of the neighbor nodes in such a network [14].

Although it has no direct security consequences, a very small value of n may cause some performance problems if the network has a high new nodes ingress rate. In order to avoid these drawbacks, a new authentication cycle has to be calculated [15].

An example in Table 1 includes the values of the second authentication cycle. There is $k^2_{\text{auth}} = [[k_M]] = [k_M]^2$ in this situation. From the value of the current tuple index δ, it can be inferred that the authenticator has carried out two successful authentications.

4.4. Authentication Protocol. A new node, A, wants to join in the network; excitation/response based mutual authentication protocol will be performed after deployment [16]. As a new node, node A can be regarded as the node first entering into the network in some certain sense, so the authenticator will be the first cycle ∇^1. Assuming node B is the authentication node to node A, which can be adopted at any cycle j, the authentication protocol between node A and node B can be described as follows:

(1) Node A produces an excitation to node B by generating a random number r_A. Then, it will send a message to node B with the following format:

$$M_1 = r_A. \tag{13}$$

(2) After receiving M_1 from node A, node B will perform the following operations:

(i) to open the first unused tuple and mark it by the current tuple index δ, so as to extract the corresponding random number r_B and the random pair $[r_B]_{k^{j-1}_{\text{auth}}}$, as well as change the status label of tuple δ from UNUSED to ASSIGNED;

(ii) to respond to an excitation from node A using the defined keyed-function with k^{j-1}_{auth} over the excitation r_A to obtain $[r_A]_{k^{j-1}_{\text{auth}}}$;

(iii) to recover its own encryption keys k^B_{enc} and the ciphers of the current authentication key to obtain $\{k^B_{\text{enc}}\}_{k^j_{\text{auth}}}$;

(iv) to restore the current cycle j of authenticator for the later synchronization with node A and send the message to node A with the following format:

$$M_2 = \left\{r_B, [r_A]_{k^{j-1}_{\text{auth}}}, \left\{k^B_{\text{enc}}\right\}_{k^j_{\text{auth}}}, j\right\}. \tag{14}$$

(3) After receiving M_2 from node B, node A will perform the following operations:

(i) to calculate the current cycle j of node B (node A is considered to be a new node, so the current cycle is 1; therefore node A needs to perform $j-1$ times hash over k_M to obtain $k^j_{\text{auth}} = [k_M]^{j-1}$ to synchronize with node B);

(ii) to check whether the response from node B is correct or not by comparing its own computation with the received value (passing the checks means node B has demonstrated the original master key and is successfully authenticated);

(iii) to calculate the imaginary part of the excitation r_B to obtain $[r_B]_{k^{j-1}_{\text{auth}}}$;

(iv) to generate its own encryption key k^A_{enc} and the ciphers of the current authentication key so as to obtain $\{k^A_{\text{enc}}\}_{k^j_{\text{auth}}}$;

(v) to send the message to node B with the following format:

$$M_3 = \left\{[r_B]_{k^j_{\text{auth}}}, \left\{k^A_{\text{enc}}\right\}_{k^j_{\text{auth}}}\right\}. \tag{15}$$

(4) Finally, after receiving M_3 from node A, node B will compare each response of the authenticator in use with the status label ∇^j_k:

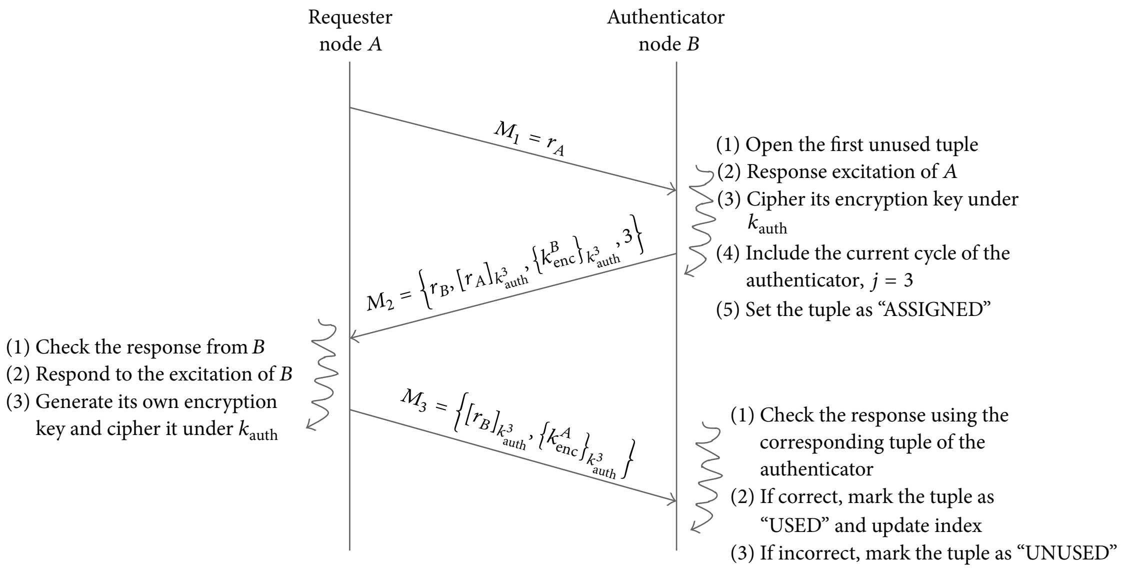
(i) if they are equal, the status label ∇^j_k is changed from ASSINGED to USED, and the new joining node is authenticated to access the network;

(ii) if they are not equal, the status label is changed to UNUSED again, which means the new joining node fails in authentication and is not allowed to access the network.

The whole information exchange process can be summarized as in Figure 2.

Being observed in steps (2) and (3), this exchange process provides simple key establishment procedures and effective transfers of the appropriate encryption key k^A_{enc} or k^B_{enc} to the corresponding groups.

Figure 2 shows a full implementation example of the proposed authentication scheme. Node A and node B are regarded as the requester and the authenticator, respectively. The authentication parameters of node B are ∇^3 and $\delta = 2$, which means that the authenticator is in the third cycle and two tuples have been successfully used. Node A is a new node, so its authenticator is in the first cycle.



Requester node A
Authenticator node B
$M_1 = r_A$
(1) Open the first unused tuple
(2) Response excitation of A
(3) Cipher its encryption key under k_{auth}
(4) Include the current cycle of the authenticator, $j = 3$
(5) Set the tuple as "ASSIGNED"
$M_2 = \{r_B, [r_A]_{k^3_{\text{auth}}}, \{k^B_{\text{enc}}\}_{k^3_{\text{auth}}}, 3\}$
(1) Check the response from B
(2) Respond to the excitation of B
(3) Generate its own encryption key and cipher it under k_{auth}
$M_3 = \{[r_B]_{k^3_{\text{auth}}}, \{k^A_{\text{enc}}\}_{k^3_{\text{auth}}}\}$
(1) Check the response using the corresponding tuple of the authenticator
(2) If correct, mark the tuple as "USED" and update index
(3) If incorrect, mark the tuple as "UNUSED"

Figure 2: Authentication protocol running example.

Table 2: Summarization of the simulation parameters (CH: cluster head; SN: sensor node).

Parameters	Values
Number of nodes	100, 101, 102, . . . , 500
Area size (m^2)	500 × 500
Wireless bandwidth (Mbps)	2
Simulation duration (sec)	300
Traffic source	CBR
Mobility speed (m/s)	0, 1, 2, . . . , 25
Initial energy (J)	CH = 50, SN = 5
Initial V_{BP} (J)	CH = 500, SN = 50
Radio range (m)	CH = 150, SN = 50
Number of internal attackers	90% of attackers
Number of external attackers	10% of attackers
Number of CHs	6% of nodes

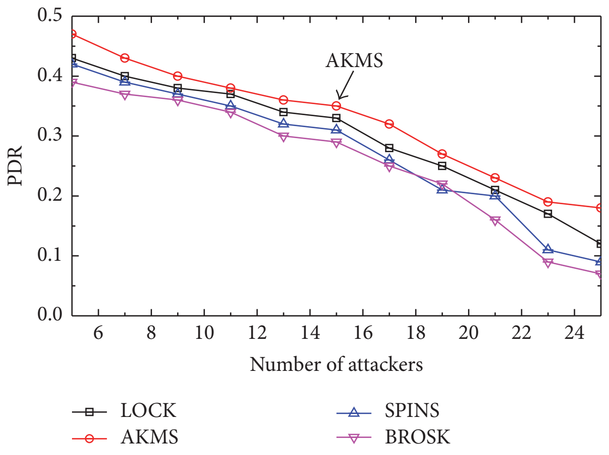


Figure 3: Packet delivery ratio.

5. Performance Evaluations

In this section, the performance of AKMS will be evaluated and analyzed in terms of the average packet delivery rate, the average energy consumption, and the networking success rate with different types of attackers.

5.1. Simulation Settings. The performance of AKMS proposed in this paper is assessed by NS2 [17]. The simulations have been carried out 20 times in different scenarios with the results being averaged for each [18]. The simulation parameters are shown in Table 2.

5.2. Simulation Results. The number of nodes remains 200 with the number of attackers being 5, 10, 15, 20, and 25, respectively [19]. Firstly, the average packet delivery rate (PDR) of AKMS is simulated compared with that of LOCK, SPINS, and BROSK, and the results are shown in Figure 3. Because of the bidirectional malware detection technology to eliminate malicious node cluster members and CH, AKMS will reduce the error packets effectively, so as to be able to send more legitimate packets to the destination compared to other schemes. Besides, AKMS adopts multipath propagation routing technology to eliminate the selective forwarding attacks, which makes the PDR even higher.

Secondly, the average energy consumption of all the nodes is measured during transmission, including the energy consumption by sending, receiving, and calculating. Figure 4 shows the average energy consumption comparisons of AKMS, LOCK, SPINS, and BROSK. When the number of attackers is growing, the average energy consumption will also increase. This is because the increasing attackers will cause more error packets. CH will filter out error packets based on AKMS to avoid the spreading of the packets from attackers throughout the network, so as to reduce the energy consumption. In LOCK, the CH node must initiate the key updating if it is captured. A new CH is selected as the new BS,

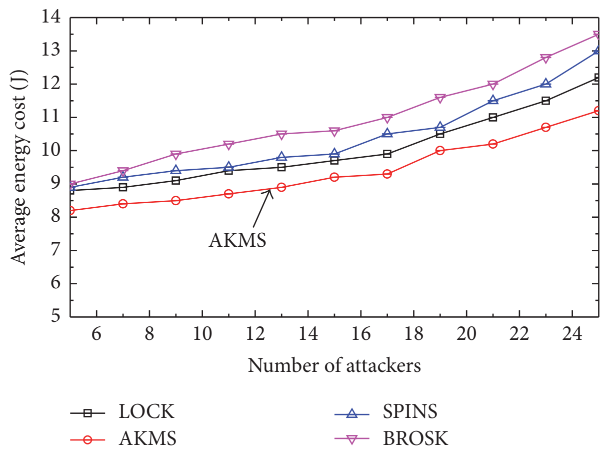
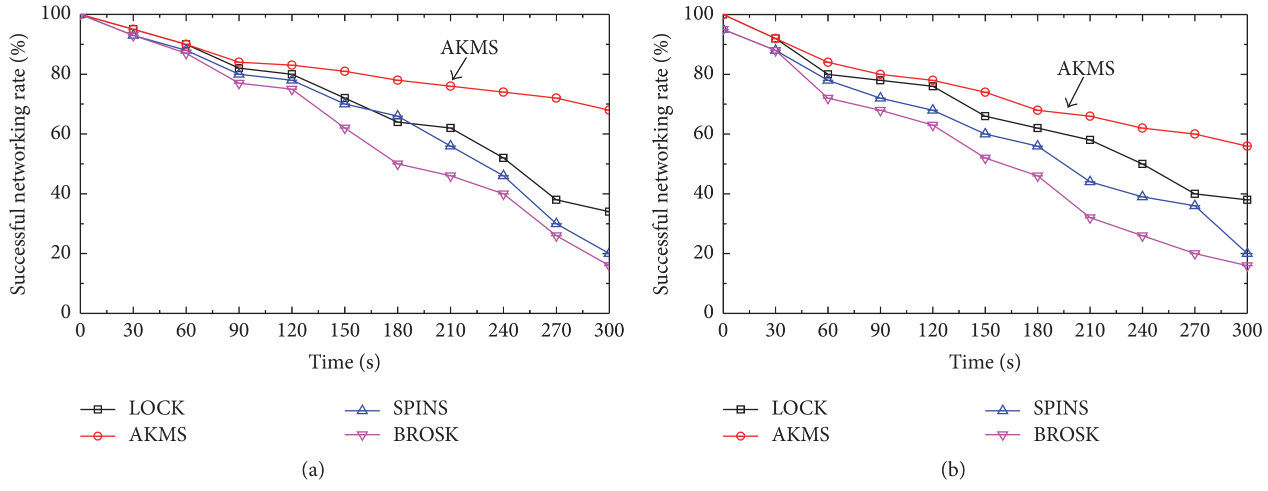


Figure 4: Energy consumption.



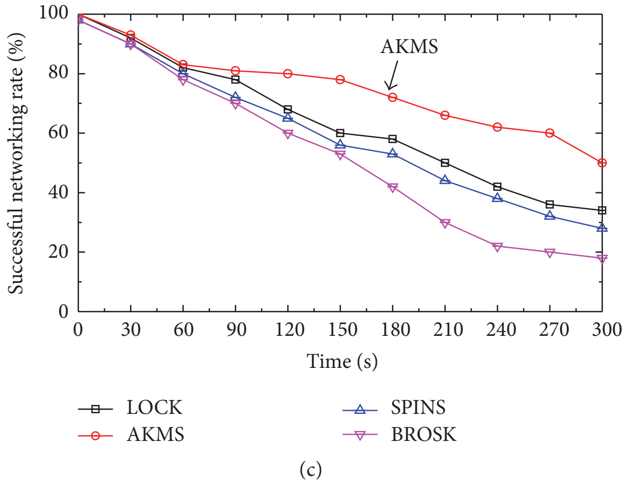


Figure 5: (a) Successful networking rate with 25 static attackers. (b) Successful networking rate with 25 mobile attackers. (c) Successful networking rate with static and mobile attackers.

and it will distribute new keys to its cluster members with the help of key generation node (KGN) [20], which will consume more energy.

Thirdly, the impact of network resilience ability has been analyzed and evaluated by the percentage of successful networking in the network of 500 nodes with 25% being the attacker. Three types of attack scenarios are set with only static attackers, only mobile attacker, and combination of both, and the simulating results are shown in Figures 5(a), 5(b), and 5(c), respectively. AKMS will detect the malicious nodes and exclude them from the network to avoid them participating in the network activities. Moreover, network environment variation based dynamic key scheme makes it difficult for the attackers to capture node. Even if the nodes are compromised by attackers, it cannot affect the entire network with AKMS.

Figure 5(a) shows the successful networking rate in the presence of 25 static attackers. In LOCK, the key updating is initiated only if the node-capturing rate reaches the Network Resilience Point (Nc). If a certain node in the network is attacked before the key updating, it will be further used by the attacker to destroy the rest of the nodes in the network. Therefore, the proportion of the mobile nodes will be decreasing rapidly. AKMS, however, is able to capture the key that is hidden before distribution or stored in each node in the cluster. Once these keys are captured, the attacker will further attempt to compromise more nodes until AKMS reinitiates the key updating. Since AKMS support mobility, the neighbors of any attacker may move to another cluster, so the performance of AKMS will be better than other schemes, especially in mobile WSN.

Figure 5(b) shows the successful networking rate in the presence of 25 mobile attackers. The attackers in the network are moving with different speed. Generally, the attacker with the maximum speed is able to attack the most nodes by moving from one to another cluster quickly and continuously launching attacks before being recognized and separated. Simulating results and analysis both indicate that the performance of AKMS is better than that of LOCK, SPINS, and BROSK, because AKMS can identify the malicious nodes and isolate them from the network at the same time.

Figure 5(c) shows the successful networking rate in the presence of 10 mobile and 15 static attackers. From the simulation it can be seen that the static attackers are able to be identified before the network becomes stable; the mobile attackers will attack by moving from one to another cluster. AKMS will perform better because of the mobility-support characteristics.

6. Conclusions

The resource-constrained essence of WSN makes the attack threat, security system, and algorithm quite different from those in traditional wireless network. In this paper, a lightweight authentication and key management protocol AKMS has been proposed for wireless sensor networks. It uses the symmetric cryptographic primitives with keyed-hash functions (HMAC) and bidirectional encryption algorithm to provide message confidentiality and authenticity for WSN and reduces the encryption overhead to the minimum as well with just a few bytes to be performed for once per authentication attempt. Simulation results show that the proposed scheme AKMS will provide more efficient security with less energy consumption, control overhead, and packet loss rate than other typical schemes, and the advantages will become remarkable with the number of nodes, attackers, and cycles increasing. Moreover, for the condition that there are mobile sensors in the network, the proposed scheme AKMS performs quite well compared to LOCK, SPINS, and BROSK. Future research will focus on the way to resist various attacks and robust routing in ubiquitous communication network.

Notations

k_M:	Master key in the whole network
k^A_{enc}:	Encryption key of node A
$\{M\}_k$:	Encryption of message M with key k
$[M]^i$:	Message M is hashed i times without key
∇^i_j:	The jth tuple of the ith cycle authenticator
$[M]$:	Hash of message M
V_{BP}:	Virtual battery power
$[M]^i_k$:	Message M is hashed i times with key k
$[M]_k$:	Hash of message M with key k (HMAC)
k^j_{auth}:	Authentication key of the jth authentication cycle; that is, $k^j_{\text{auth}} = [k_M]^j$.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61302074, 61571181), Natural Science Foundation of Heilongjiang Province (QC2013C061), Modern Sensor Technology Research and Innovation Team Foundation of Heilongjiang Province (2012TD007), Postdoctoral Research Foundation of Heilongjiang Province (LBH-Q15121), and Postgraduate Innovation Research Foundation of Heilongjiang University (YJSCX2016-019HLJU).

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] Z. Chen, M. He, W. Liang, and K. Chen, “Trust-aware and low energy consumption security topology protocol of wireless sensor network,” *Journal of Sensors*, vol. 2015, Article ID 716468, 10 pages, 2015.

[3] S. Verma and Prachi, “Key pre-distribution scheme for WSNs,” *Ad-Hoc and Sensor Wireless Networks*, vol. 23, no. 1-2, pp. 47–67, 2014.

[4] S. Bag, “A new key predistribution scheme for grid-group deployment of wireless sensor networks,” *Ad Hoc and Sensor Wireless Networks*, vol. 27, no. 3-4, pp. 313–329, 2015.

[5] M. Eltoweissy, M. Moharrum, and R. Mukkamala, “Dynamic key management in sensor networks,” *IEEE Communications Magazine*, vol. 44, no. 4, pp. 122–130, 2006.

[6] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, “Spins: security protocols for sensor networks,” *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[7] B. C. Lai, D. D. Hwang, S. P. Kim, and I. Verbauwhede, “Reducing radio energy consumption of key management protocols for wireless sensor networks,” in *Proceedings of the ACM International Symposium on Low Power Electronics and Design (ISLPED ’04)*, pp. 351–356, Newport Beach, Calif, USA, 2004.

[8] O. D. Mohatar, A. F. Sabater, and J. M. Sierra, “A light-weight authentication scheme for wireless sensor networks,” *Ad Hoc Networks*, vol. 9, no. 5, pp. 727–735, 2011.

[9] F. Ishmanov, A. S. Malik, S. W. Kim, and B. Begalov, “Trust management system in wireless sensor networks: design considerations and research challenges,” *Transactions on Emerging Telecommunications Technologies*, vol. 26, no. 2, pp. 107–130, 2015.

[10] A. Yener and S. Ulukus, “Wireless physical-layer security: lessons learned from information theory,” *Proceedings of the IEEE*, vol. 103, no. 10, pp. 1814–1825, 2015.

[11] J. Rabaey and J. Ammer, “Distributed framework for correlated data gathering in sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 57, no. 1, pp. 578–593, 2010.

[12] K. C. Egemen, B. Dan, and D. Amit, “Modelling communication network challenges for future internet resilience, survivability, and disruption tolerance: a simulation-based approach,” *Telecommunication Systems*, vol. 2, no. 6, pp. 751–768, 2013.

[13] L. Lamport, “Password authentication with insecure communication,” *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[14] O. Bazan and M. Jaseemuddin, “A survey on MAC protocols for wireless adhoc networks with beamforming antennas,” *IEEE Communications Surveys and Tutorials*, vol. 14, no. 2, pp. 216–239, 2012.

[15] H. Wang, J. Lai, M. Hu, and Y. Liang, “Research on key technologies for implementing network security situation awareness,” *Geomatics and Information Science of Wuhan University*, vol. 33, no. 10, pp. 995–998, 2008.

[16] P.-H. Lin, S.-H. Lai, S.-C. Lin, and H.-J. Su, “On secrecy rate of the generalized artificial-noise assisted secure beamforming for wiretap channels,” *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 1728–1740, 2013.

[17] D. Lucas and J. Joel, “A survey on cross-layer solutions for wireless sensor networks,” *IEICE Transaction on Journal of Network and Computer Applications*, vol. 5, no. 34, pp. 523–534, 2011.

[18] C. Wang, X.-Y. Shi, and Z.-H. Niu, “The research of the promotion for ECDSA algorithm based on Montgomery-form ECC,” *Journal on Communications*, vol. 31, no. 1, pp. 9–13, 2010.

[19] K. Okeya and K. Sakurai, “A scalar multiplication algorithm with recovery of the y-coordinate on the montgomery form and analysis of efficiency for elliptic curve cryptosystems,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 85, no. 1, pp. 84–93, 2002.

[20] G. Anastasi and M. Conti, “Data collection in sensor networks with data mules: an intergrade simulation analysis,” *IEEE Symposium on Computers and Communications*, vol. 3, no. 7, pp. 1096–1102, 2012.